Locker Security System

CS122A: Fall 2017

Benjamin Long

Table of Contents

[**Introduction**](#_n9o44neiquk5) **2**

[**Hardware**](#_vqmbsuo5p330) **2**

[Parts List](#_xtqh2tzibtrr) 2

[Block Diagram](#_8m8fex2vr9el) 2

[Pinout (For each microcontroller/processor)](#_8yptvhjzojq8) 3

[**Software**](#_w9543qe124on) **3**

[**Implementation Reflection**](#_yfnwo0ojgf6a) **4**

[Milestone](#_2j4av972xaj8) 4

[Completed components](#_tp9dnsse4u9p) 4

[Incomplete components](#_99jbzhpy2eks) 4

[**Youtube Links**](#_qj6ffofi3ean) **4**

[**Testing**](#_stprv61uxgvu) **4**

[**Known Bugs**](#_grlzpb6vy2cq) **5**

[**Resume/Curriculum Vitae (CV) Blurb**](#_rvkpcxu3jywt) **5**

[**Future work**](#_qbv6f31drpex) **5**

[**Extra Credit**](#_54hn6ztqvlw9) **5**

[**References**](#_1txxevdbm0qb) **5**

[**Appendix**](#_i5qunlt5l708) **6**

# Introduction

The Locker Security System is a quick and reliable way to setup a secure locker whenever you go to the gym. Most lockers require a physical lock and key. This locker system requires nothing but a user created passcode.

Users are able to reserve lockers for a one time storage use. The system allows users to create a password pattern via joystick and assign that pattern to any available locker. After setting up their locker, the user can unlock their locker by just selecting their locker number on the display, and inputting their pattern. If inputted correctly, their locker should automatically open. This locker security system can be deployed in commercial gyms, high schools, beaches, and many outdoor recreation centers.

Insert a picture of your project (Your completed project).

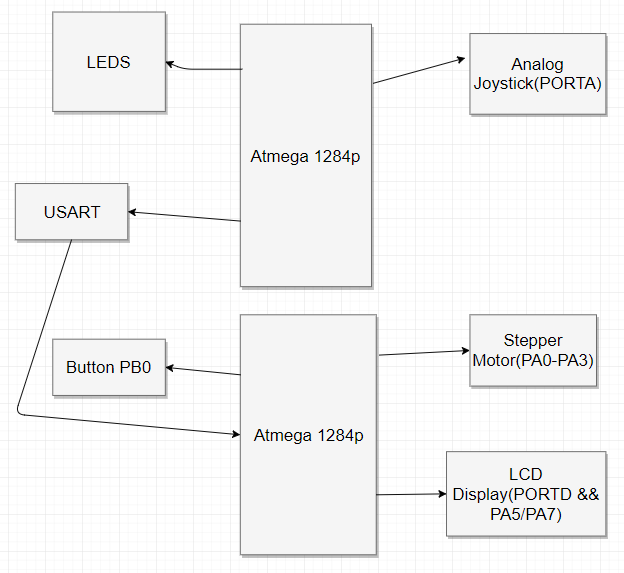
# Hardware

## Parts List

The hardware that was **used** in this project is listed below. The equipment that was not taught in this course has been bolded. *Include part numbers when available*.

|  |  |  |  |
| --- | --- | --- | --- |
| Part | Part # | Quantity | Price (optional) |
| ATMega1284 | ATMega1284 | 2 |  |
| Analog Joystick |  | 1 |  |
| 16x2 LCD Display | ... | 1 | ... |
| Stepper Motors |  | 1 |  |
| LEDs |  | 4 |  |
| Button |  | 1 |  |
|  |  | **Total** |  |

## Block Diagram

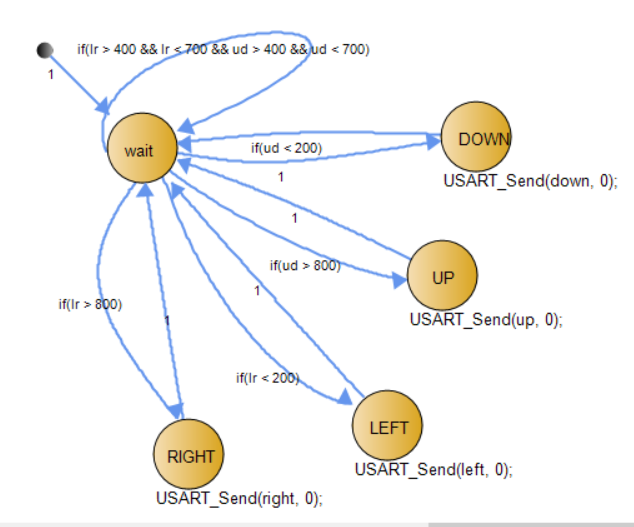


## Pinout (For each microcontroller/processor)

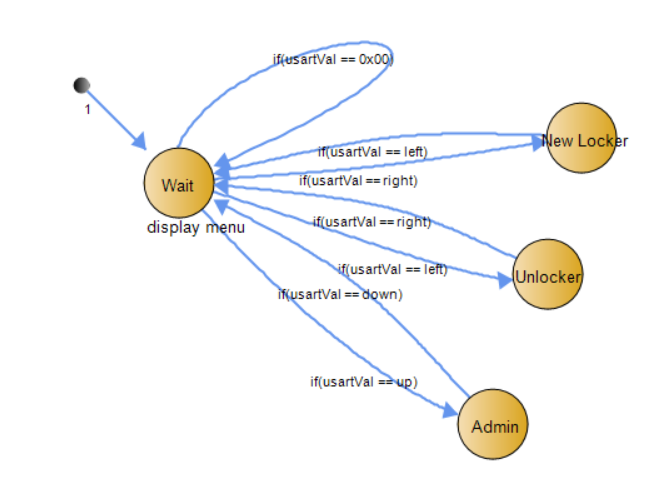
* Microcontroller 1
  + PA0/PA1 : Analog Joystick
  + PD0/P1 : USART Communication
  + PB0-PB3 : LEDs
* Microcontroller 2
  + PA0-PA3 : Stepper Motor
  + PB0 : Select Button
  + PD0/PD1 : USART Communication
  + PA5/PA7/PORTC : 16x2 LCD Display

# Software

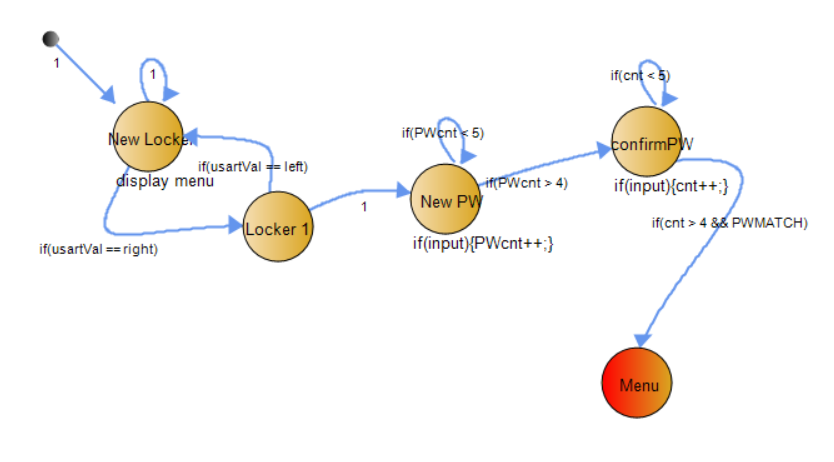
The software designed for this project was implemented using the PES standard. The overall design as a task diagram is included below.



This diagram shows how the analog joystick is read and how the input is sent to the second microcontroller via USART. The state remains in wait until a value from the horizontal and vertical axis is read in. The wait state then transitions to the appropriate state, and then sends the input to the second microcontroller via USART.



This is a basic diagram of the LCD Display menu. The SM waits until USART receives joystick information, and then transitions to the according states. If a select button is pressed within New Locker, Unlocker, or Admin, the state transitions to either Password setup or Password Confirmation states.



This diagram is a similar to other state machines that involve password creation, password input, and password confirmation. A locker is chosen, and reads in joystick directions as password entries. These password entries are stored into arrays that correspond to their appropriate locker numbers.

# Implementation Reflection

I feel that my project was moderately difficult. The software development of my project took much more time than I had anticipated. Majority of my time was spent debugging software code and hardware issues.

I really enjoyed seeing the development of my user menu. Each time I completed a working menu option, I felt my project coming to life which brought me great satisfaction. If I were to do the project again, I would consider spending more time during the planning process before starting my project. If I had thought and planned more carefully, my project would have been more well-polished.

## Milestone

My milestone was to implement all of the code and get my security system functions working properly. This includes the ability to dynamically assign lockers and password patterns to people profiles. When a correct pattern is inputted, a motor should activate to release a locker door. The display and user interface should also be fully implemented as well. If a user forgets their pattern, then they must select the “Forgot password” option and seek help from an admin(I actually changed the “Forgot password” option to an “admin” option). If a password has been inputted incorrectly too many times, then the locker will freeze and not open.

I reached my milestone late at the end of week 8, so I never got checked off. Although I eventually reached my milestone, I still believe I fell short due to unexpected hardware, family, and personal issues. If I had taken into more consideration of time for unexpected situations, I could have progressed my project further.

## Completed components

How much of your proposed project did you complete? Did you get to 70 - 80, to 80 - 90, 90 - 100, or did you fall short? Why?

I believe I have reached just under my 70-80 goal. I completed the menu functions which allow users to setup new passwords for new lockers, unlock previously setup lockers, and call admin to help users unlock their lockers. If incorrect passwords are entered, then the locker does not open. The LCD Display is fully completed. It displays information corresponding to user inputs received from the joystick via USART. The Joystick inputs are read in correctly and are sent through USART to the second microcontroller. The Stepper motors open the locker doors correctly, but do not close properly.

## Incomplete components

I was unable to complete my locker motor function properly. The locker motor does not close locker door properly. Instead of reversing 90 degrees, the locker door swings back around 270 degrees to get back to the closed position. I was able to complete the stepper motor lab without a problem. However, as I applied the motor to my project, I was unable to get my motor to move in reverse. Although the code for implementing several lockers is the same, I only have locker 1 implemented.

A difficulty to note in the development of my project is using USART to communicate my joystick with the LCD display that are used on two different microprocessors. Getting my LCD display to correspond to the data received via USART took a considerable amount of time to do. In the end, I was able to synch the analog inputs to my display correctly.

# Youtube Links

https://www.youtube.com/watch?v=YEtpcRzwbiQ

# Testing

Include a sub header for each component/feature.

What did you end up doing to test your project? Who did you have test it?

Write our the test cases that you used, and the results of those test cases.

I allowed non-engineering students to test out my project. They attempted to setup and unlock lockers within the system and reported back their observations. Some of these observations referred to known bugs and glitches such as improper state transitions and flickering displays.

* Joystick Test
  + I tested the joystick input by using LEDs to represent different directions
  + If the joystick is pushed up, then the ‘up’ LED should light up.
  + If the joystick is in neutral, then no LEDs should light up.
    - The joystick test was successful
* LCD Display Test
  + I tested this display out by outputting different menu prompts
  + If the user moves the joystick up, left, or right, then the menu should scroll to its appropriate menu prompt
    - The display test was successful
* USART Test
  + I tested this communication by using a basic button and led test. When a button connected to a microcontroller 1 is pressed, USART transmits data to microcontroller 2. When microcontroller 2 receives this specific data, LEDs connected to microcontroller 2 light up.
    - The USART test was successful

# Known Bugs

* After unlocking a locker, the locker system sometimes gets stuck in the locker selection menu instead of returning to the main menu. This bug was discovered by my non-engineer friend who was testing my project out at the time. My guess is that I am not transitioning between my states properly somewhere in my code. To fix this bug, I would have to step through my code and see what the sequence of my states are.
* Another bug is that the LCD display seems to not be as smooth as it could be. I believe this is due to looping back into a state over and over again that outputs to the LCD display. As a result, the display might be constantly updated, causing the “flickering” within the display. To fix this bug, I would redesign my states so that the LCD display outputs only when needed to be.

# Resume/Curriculum Vitae (CV) Blurb

December 2017 – Locker Security System

This locker system implements the communication skills of USART between two Atmega1284p microcontrollers. The system takes user input from an analog joystick, and converts the analog to digital input, and transmits the input to a second microcontroller. The second microcontroller receives this data, and displays on a 16x2 LCD display menu options, password prompts, and user prompts appropriate to the user actions. The security system stores passcodes within arrays and compares these password arrays to user passcode inputs for unlocking purposes.

# Future work

In this section pretend for a moment that you were going to continue working on the project.

* What would be the next feature you would add?
* How would you proceed?
* What would you do to design a case?
* Could you create a custom PCB for your project to reduce the size?

If you did something similar to the project you proposed in your Company Research Report, will you be using it to apply to that company?

For future work, I would implement a time limit for each locker use. This time limit prevents users from reserving lockers for long periods of time so that other users may have the chance to access these lockers. To design this, I would implement a timer that sets a flag whenever the timer has run out. This flag notifies the system to lock that specific locker, so that the user must request admin help to unlock their locker. This system will create an incentive for users to not hog lockers so that others may use them.

This project is similar to that of my project report. It is suitable for a company project application.

# Extra Credit

Include a link to a public DIY for your project: <link>

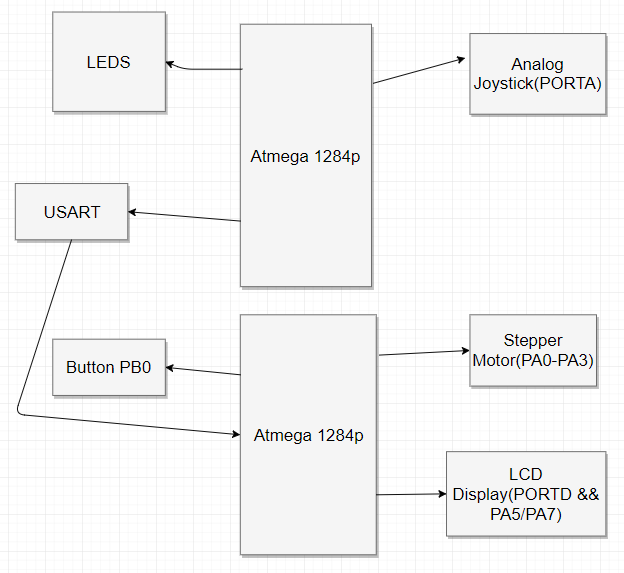
# References

If you used code or ideas from another source (including inspiration from a video or another product) please list those sources here.

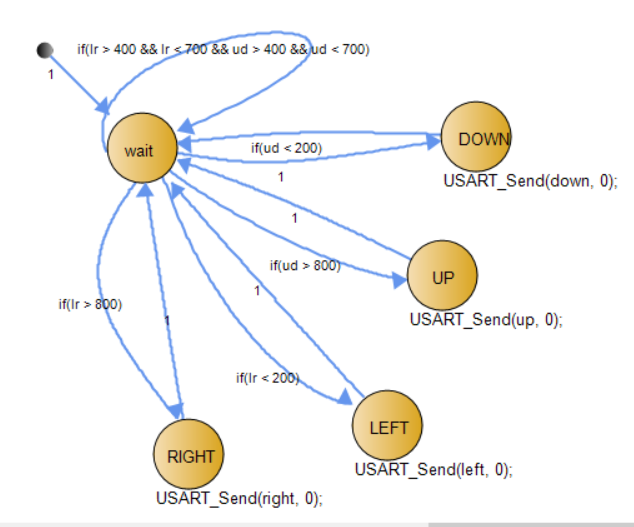
# Appendix

Include images of all of the SM’s that you have designed and any other work that you think is relevant to this project.

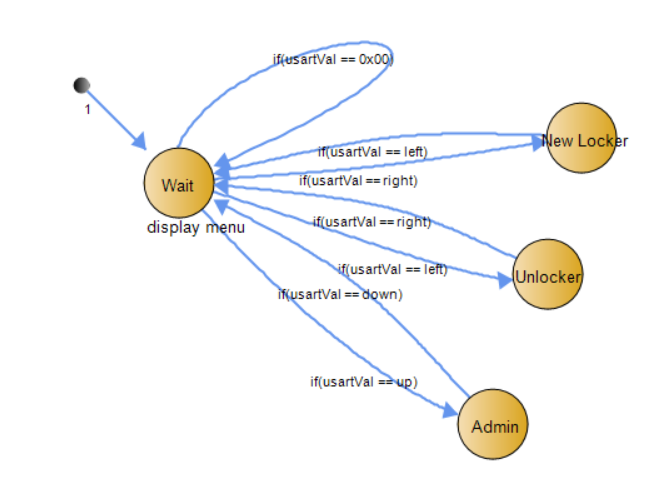
BLOCK DIAGRAM:



JOYSTICK SM:



MENU SM:



LOCKER CREATION/PASSWORD SM:

